Statistical Modelling III

HES 505 Fall 2024: Session 23

Carolyn Koehn

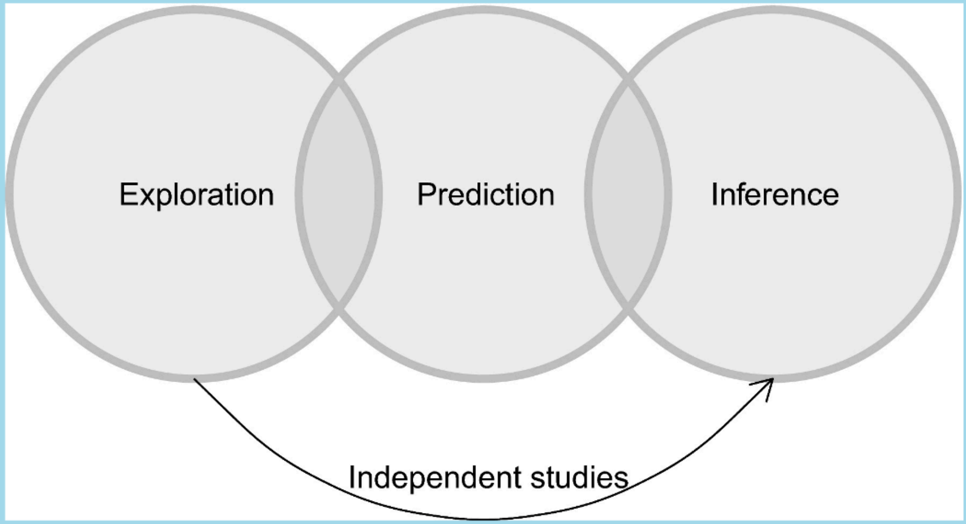
Objectives

By the end of today you should be able to:

- Articulate three different reasons for modeling and how they link to assessments of fit
- Describe and implement several test statistics for assessing model fit
- Describe and implement several assessments of classification
- Describe and implement resampling techniques to estimate predictive performance

The 3 Faces of Models

Best Model for What?



from Tradennick et al. 2021

- **Exploration:** describe patterns in the data and generate hypotheses
- **Inference:** evaluate the strength of evidence for some statement about the process
- **Prediction:** forecast outcomes at unsampled locations based on covariates

The Importance of Model Fit

- The general regression context:

$$\hat{y} = \mathbf{X}\hat{\beta}$$

- **Inference** is focused on robust estimates of $\hat{\beta}$ given the data we have
- **Prediction** is focused on accurate forecasts of $\hat{y}$ at locations where we have yet to collect the data

Inference and Presence/Absence Data

- $\hat{\beta}$ is conditional on variables in the model **and** those not in the model

```r
nsamp <- 1000
df <- data.frame(x1 = rnorm(nsamp,0,1),
                 x2 = rnorm(nsamp,0,1),
                 x3 = rnorm(nsamp,0,1))

linpred <- 1 + 2*df$x1 -0.18*df$x2 -3.5*df$x3
y <- rbinom(nsamp, 1, plogis(linpred))
df <- cbind(df, y)

mod1 <- glm(y~x1 +x2, data=df, family="binomial")
mod2 <- glm(y~x1 +x2 + x3, data=df, family="binomial")
```

Inference & Presence/Absence Data

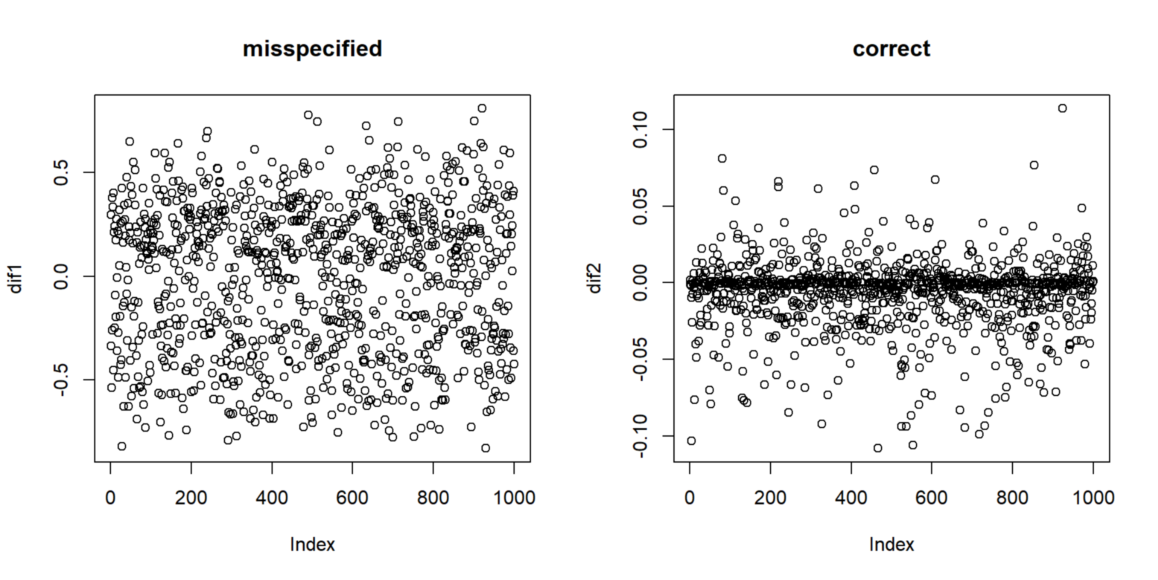
```r
coef(mod1)
```

```
(Intercept)          x1
x2
  0.4460172   0.9041635
-0.1018929
```

```r
coef(mod2)
```

```
(Intercept)          x1
x2          x3
   1.175262    2.298450
-0.307852   -3.774803
```

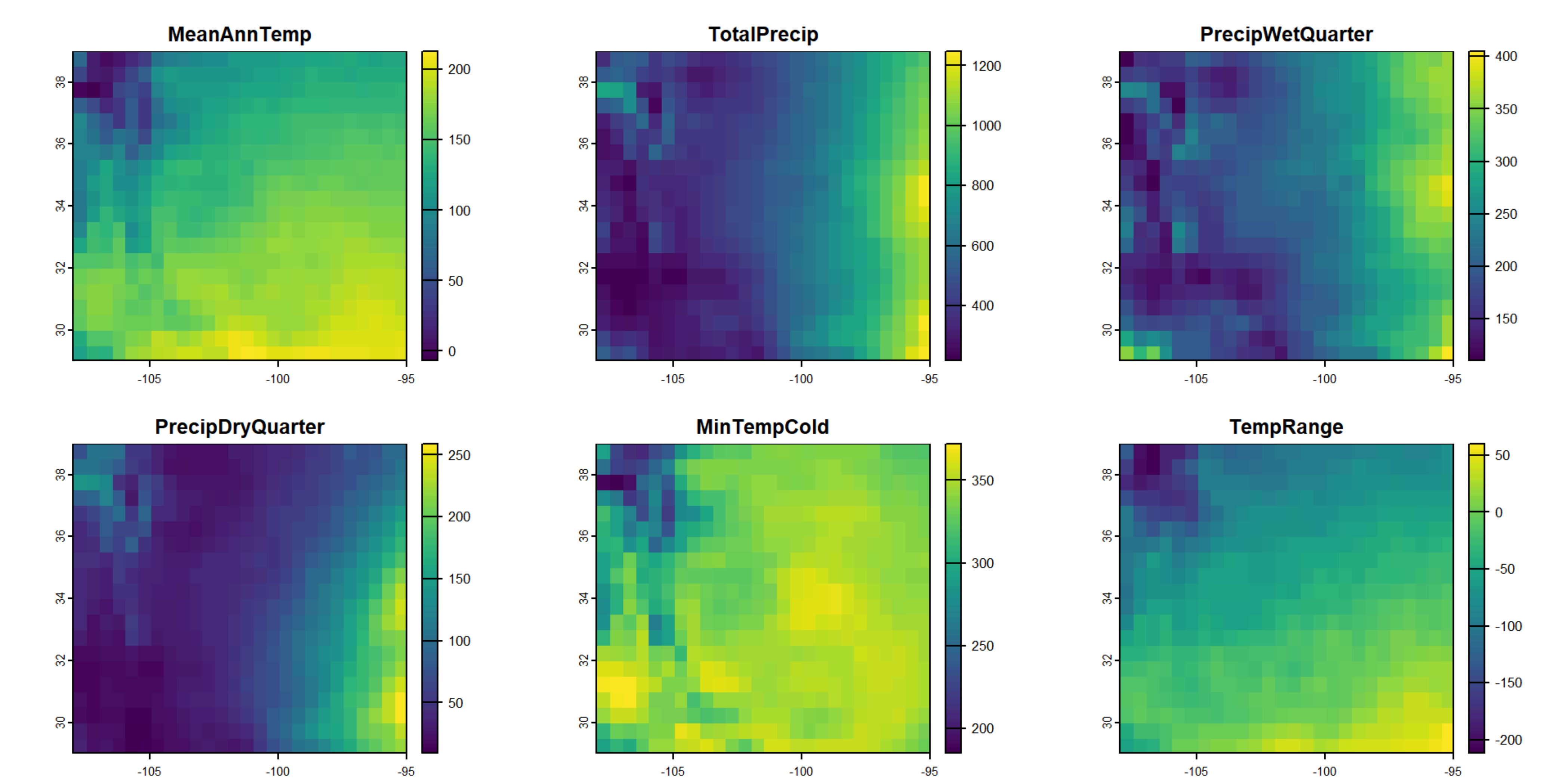
```r
prd1 <- predict(mod1, df, "response")
dif1 <- plogis(linpred) - prd1
prd2 <- predict(mod2, df, "response")
dif2 <- plogis(linpred) - prd2
```



> Inferring coefficient effects requires that your model fit the data well

Assessing Model Fit

Back to our simulated data



Back to our simulated data

```r
base.path <- "/opt/data/data/presabsexample/" #sets the path to the root di

pres.abs <- st_read(paste0(base.path, "presenceabsence.shp"), quiet = TRUE)
pred.files <- list.files(base.path,pattern='grd$', full.names=TRUE) #get th

pred.stack <- rast(pred.files) #read into a RasterStack
names(pred.stack) <- c("MeanAnnTemp", "TotalPrecip", "PrecipWetQuarter", "P
plot(pred.stack)
pred.stack.scl <- scale(pred.stack)
pts.df <- terra::extract(pred.stack.scl, vect(pres.abs), df=TRUE)
pts.df <- cbind(pts.df, pres.abs$y)
colnames(pts.df)[8] <- "y"
```

Using Test Statistics

- R^2 for linear regression:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

$$SS_{res} = \sum_i (y_i - f_i)^2$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$

- Perfect prediction ($f_i = y_i$); $SS_{res} = 0$; and $R^2 = 1$
- Null prediction (Intercept only) ($f_i = \bar{y}$); $SS_{res} = SS_{tot}$; and $R^2 = 0$
- No direct way of implementing for logistic regression

Pseudo- R^2

$$R_L^2 = \frac{D_{null} - D_{fitted}}{D_{null}}$$

- Cohen's Likelihood Ratio
- Deviance (D), the difference between the model and some hypothetical perfect model (lower is better)
- Challenge: Not monotonically related to p
- Challenge: How high is too high?

Cohen’s Likelihood Ratio

```
logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])
logistic.simple <- glm(y ~ MeanAnnTemp + TotalPrecip,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

# Pseudo-R^2
with(logistic.rich,
     null.deviance - deviance)/with(logistic.rich,
                                    null.deviance)
```

```
[1] 0.4495966
```

```
with(logistic.simple,
     null.deviance - deviance)/with(logistic.simple,
                                    null.deviance)
```

```
[1] 0.4567641
```

Pseudo- R^2

$$R^2_{CS} = 1 - \left(\frac{L_0}{L_M}\right)^{(2/n)}$$
$$= 1 - \exp^{2(ln(L_0) - ln(L_M))/n}$$

- Cox and Snell R^2
- Likelihood (L), the probability of observing the sample given an assumed distribution
- Challenge: Maximum value is less than 1 and changes with n
- Correction by Nagelkerke so that maximum is 1

Cox and Snell R^2

```
logistic.null <- glm(y ~ 1,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

# Cox & Snell R^2 for logistic.rich
1 - exp(2*(logLik(logistic.null)[1] - logLik(logistic.rich)[1])/nobs(logist
```

```
[1] 0.4308873
```

```
# Cox & Snell R^2 for logistic.simple
1 - exp(2*(logLik(logistic.null)[1] - logLik(logistic.simple)[1])/nobs(logi
```

```
[1] 0.4359785
```

Using Test Statistics

- Based on the data used in the model (i.e., not prediction)
- Likelihood Ratio behaves most similarly to R^2
- Cox and Snell (and Nagelkerke) increases with more presences
- Ongoing debate over which is “best”
- **Don’t defer to a single statistic**

Assessing Predictive Ability

Predictive Performance and Fit

- Predictive performance can be an estimate of fit
- Comparisons are often relative (better $\neq$ good)
- Theoretical and subsampling methods

Theoretical Assessment of Predictive Performance

Hirotugu Akaike of AIC

- Information Criterion Methods
- Minimize the amount of information lost by using model to approximate true process
- Trade-off between fit and overfitting
- Can't know the true process (so comparisons are relative)

$$AIC = -2ln(\hat{L}) + 2k$$

AIC Comparison

```r
logistic.null$formula
```

```
y ~ 1
```

```r
logistic.rich$formula
```

```
y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter
```

```r
logistic.simple$formula
```

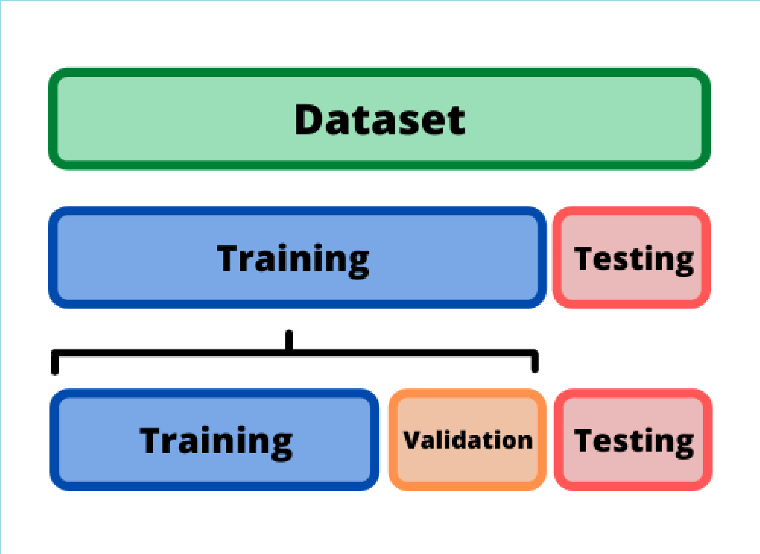
```
y ~ MeanAnnTemp + TotalPrecip
```

```r
AIC(logistic.null, logistic.rich, logistic.simple)
```

```
                df       AIC
logistic.null    1 127.37389
logistic.rich    4  77.00622
logistic.simple  3  74.10760
```

Sub-sampling Methods

- Split data into *training* and *testing*
- Testing set needs to be large enough for results to be statistically meaningful
- Test set should be representative of the data as a whole
- Validation data used to tune parameters (not always)



Subsampling your data with caret

```r
pts.df$y <- factor(ifelse(pts.df$y == 1, "Yes", "No"),
                              levels = c("Yes", "No"))
library(caret)
Train <- createDataPartition(pts.df$y, p=0.6, list=FALSE)

training <- pts.df[ Train, ]
testing <- pts.df[ -Train, ]
```

Misclassification

- Confusion matrices compare actual values to predictions

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

- True Positive (TN) - This is correctly classified as the class if interest / target.
- True Negative (TN) - This is correctly classified as not a class of interest / target.
- False Positive (FP) - This is wrongly classified as the class of interest / target.
- False Negative (FN) - This is wrongly classified as not a class of interest / target.

Confusion Matrices in R

```r
train.log <- glm(y ~ .,
                 family="binomial"
                 data=training[,2:
                 
predicted.log <- predict(train.log
                         newdata=t
                         type="res

pred <- factor(
  ifelse(predicted.log > 0.5,
                         "Yes",
                         "No"),
  levels = c("Yes", "No"))
```

```r
confusionMatrix(testing$y, pred)
```

```
Confusion Matrix and Statistics

          Reference
Prediction Yes No
       Yes   5  7
       No   26  1

               Accuracy : 0.1538
                 95% CI : (0.0586, 0.3053)
    No Information Rate : 0.7949
    P-Value [Acc > NIR] : 1.000000

                  Kappa : -0.3794

 Mcnemar's Test P-Value : 0.001728

            Sensitivity : 0.16129
            Specificity : 0.12500
         Pos Pred Value : 0.41667
         Neg Pred Value : 0.03704
             Prevalence : 0.79487
         Detection Rate : 0.12821
```

Confusion Matrices

Depends upon threshold!!

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Sensitivity (Recall)} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{FP + TN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

Confusion Matrices in R

```r
library(tree)
tree.model <- tree(y ~ . , trainin
predict.tree <- predict(tree.model
```

```r
confusionMatrix(testing$y, predict.tree)
```

```
Confusion Matrix and Statistics

          Reference
Prediction Yes No
       Yes   6  6
       No    3 24

               Accuracy : 0.7692
                 95% CI : (0.6067, 0.8887)
    No Information Rate : 0.7692
    P-Value [Acc > NIR] : 0.5882

                  Kappa : 0.4179

 Mcnemar's Test P-Value : 0.5050

            Sensitivity : 0.6667
            Specificity : 0.8000
         Pos Pred Value : 0.5000
         Neg Pred Value : 0.8889
             Prevalence : 0.2308
         Detection Rate : 0.1538
```

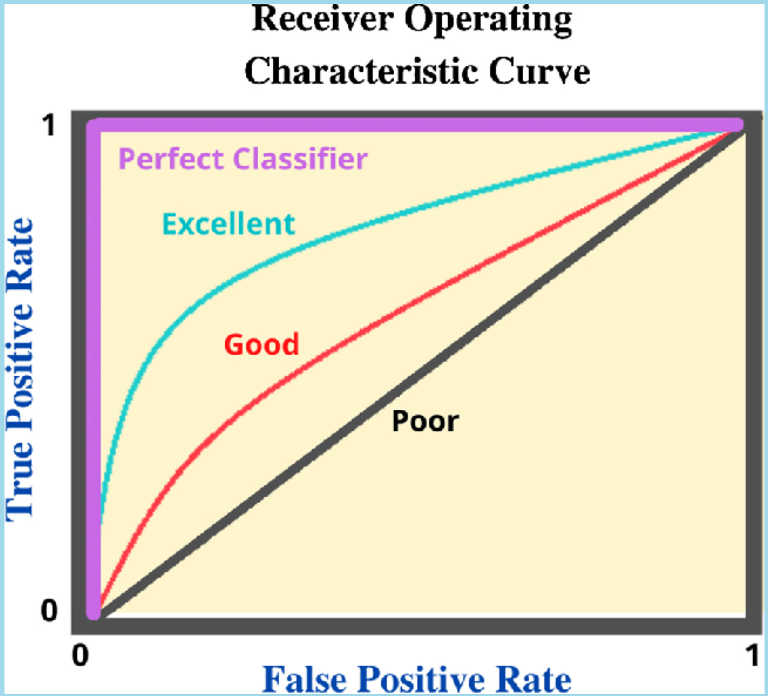
Confusion Matrices in R

```
library(randomForest, quietly = TR
class.model <- y ~ .
rf <- randomForest(class.model, da
predict.rf <- predict(rf, newdata=
```

```
confusionMatrix(testing$y, predict.rf)
```

```
Confusion Matrix and Statistics

          Reference
Prediction Yes No
       Yes   7  5
       No    4 23

               Accuracy : 0.7692
                 95% CI : (0.6067, 0.8887)
    No Information Rate : 0.7179
    P-Value [Acc > NIR] : 0.3037

                  Kappa : 0.4455

 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.6364
            Specificity : 0.8214
         Pos Pred Value : 0.5833
         Neg Pred Value : 0.8519
             Prevalence : 0.2821
         Detection Rate : 0.1795
```

Threshold-Free Methods



- Receiver Operating Characteristic Curves
- Illustrates discrimination of binary classifier as the threshold is varied
- Area Under the Curve (AUC) provides an estimate of classification ability

Criticisms of ROC/AUC

- Treats false positives and false negatives equally
- Undervalues models that predict across smaller geographies
- Focus on *discrimination* and not *calibration*
- New methods for presence-only data

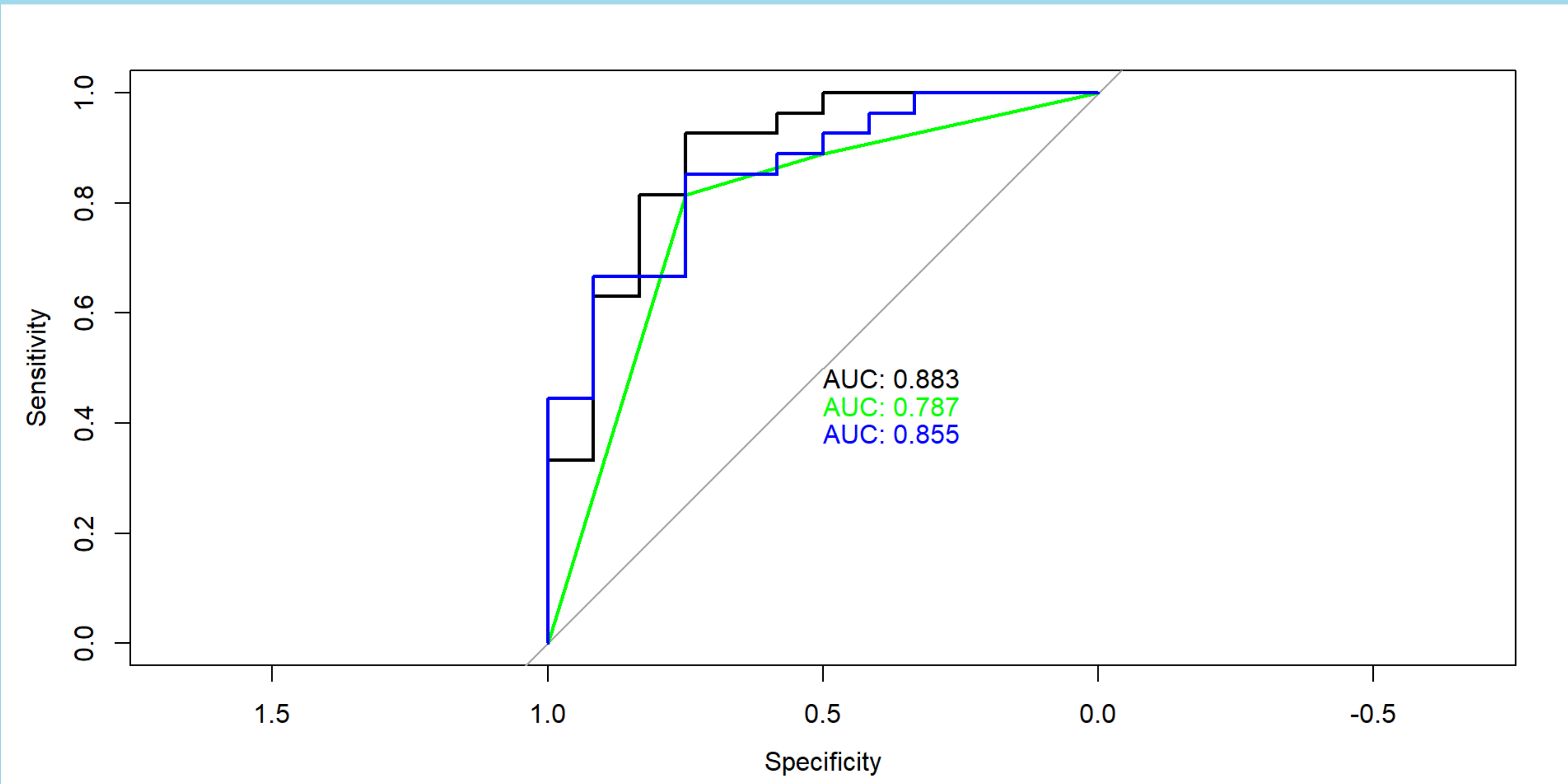
ROC in R (using pROC)

- Generate predictions (note the difference for tree and rf)

```
library(pROC, quietly = TRUE)
train.log <- glm(y ~ .,
                 family="binomial",
                 data=training[,2:8])

predicted.log <- predict(train.log,
                         newdata=testing[,2:8],
                         type="response")

predict.tree <- predict(tree.model, newdata=testing[,2:8], type="vector")[,

predict.rf <- predict(rf, newdata=testing[,2:8], type="prob")[,2]
```

ROC in R (using pROC)

```
plot(roc(testing$y, predicted.log), print.auc=TRUE)

plot(roc(testing$y, predict.tree), print.auc=TRUE, print.auc.y = 0.45, col=

plot(roc(testing$y, predict.rf), print.auc=TRUE, print.auc.y = 0.4, col="bl
```



Cross-validation

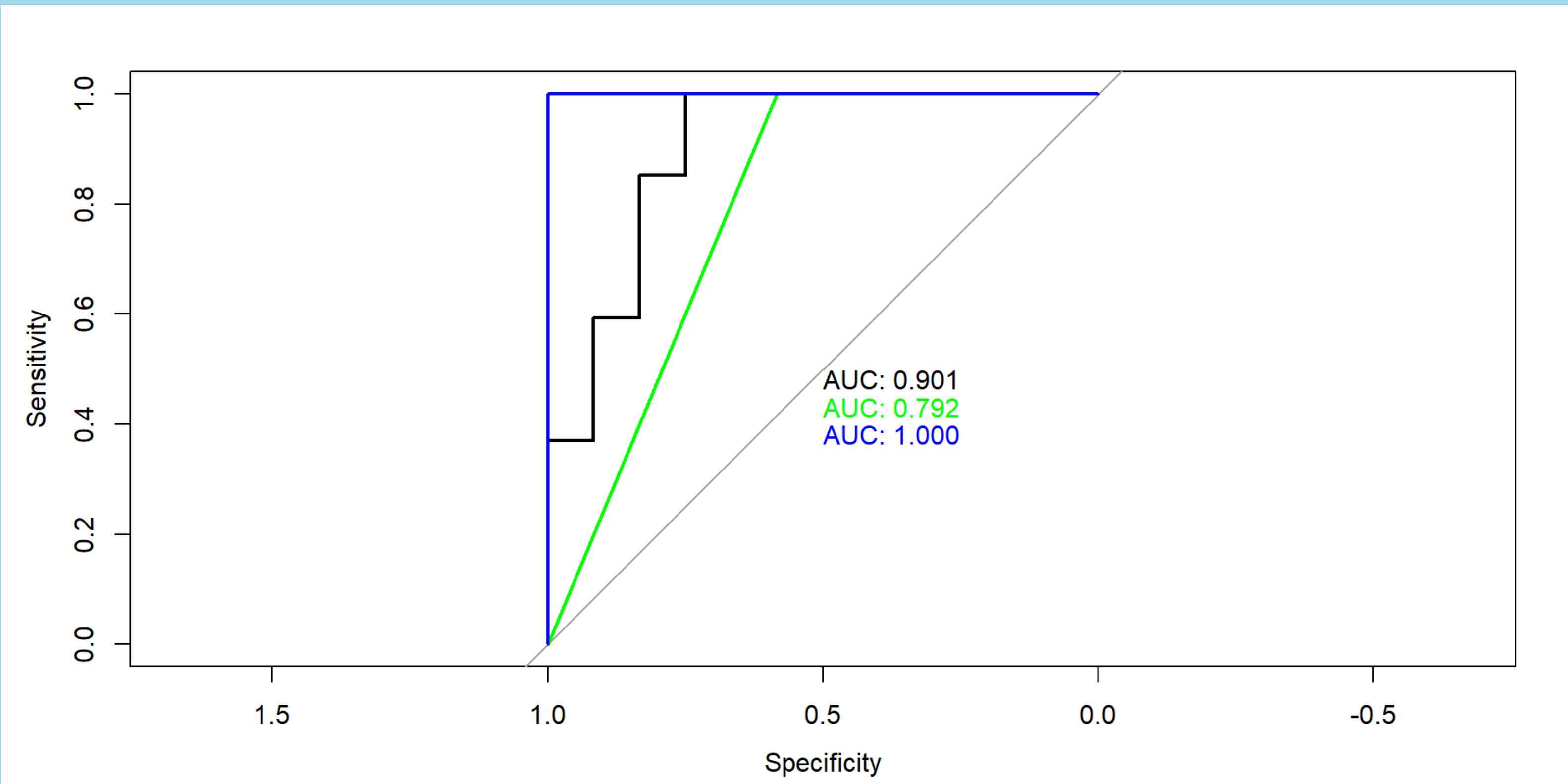
- Often want to make sure that fit/accuracy not a function of partition choice
- Cross-validation allows resampling of data (multiple times)
- K-fold - Data are split into K datasets of ~ equal size, model fit to $(K-1)(\frac{n}{K})$ observations to predict held-out set
- Leave One Out (LOO) - Model fit to n-1 observations to predict the held out observation

Crossvalidation in R using caret

```r
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary)

log.model <- train(y ~., data = pts.df[,2:8],
               method = "glm",
               trControl = fitControl)
pred.log <- predict(log.model, newdata = testing[,2:8], type="prob")[,2]

tree.model <- train(y ~., data = pts.df[,2:8],
               method = "rpart",
               trControl = fitControl)

pred.tree <- predict(tree.model, newdata=testing[,2:8], type="prob")[,2]

rf.model <- train(y ~., data = pts.df[,2:8],
```

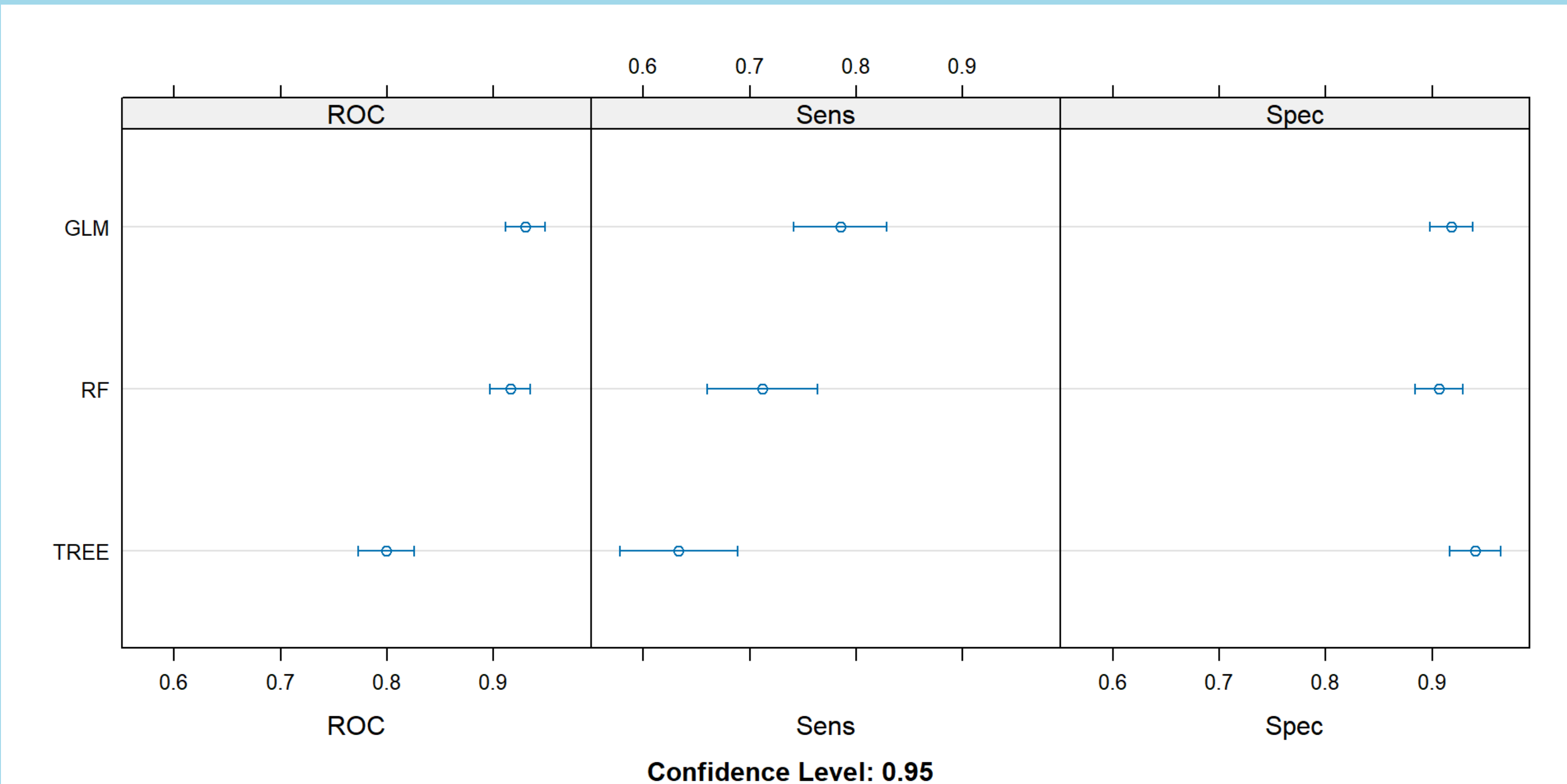
Crossvalidation in R using caret

```
plot(roc(testing$y, pred.log), print.auc=TRUE)

plot(roc(testing$y, pred.tree), print.auc=TRUE, print.auc.y = 0.45, col="gr

plot(roc(testing$y, pred.rf), print.auc=TRUE, print.auc.y = 0.4, col="blue"
```



Crossvalidation in R using caret

```r
resamps <- resamples(list(GLM = log.model,
                          TREE = tree.model,
                          RF = rf.model))
dotplot(resamps)
```



Spatial predictions

```
best.rf <- rf.model$finalModel
best.glm <- log.model$finalModel

rf.spatial <- terra::predict(pred.stack.scl, best.rf, type="prob")


glm.spatial <- terra::predict(pred.stack.scl, best.glm,type="response" )
```

Spatial predictions